REMARKS

This application has been carefully reviewed in light of the Office Action dated May 14, 2007. Claims 1-5 and 7-21 remain in this application. Claims 1, 11 and 16 are the independent Claims. It is believed that no new matter is involved in the amendments or arguments presented herein. Reconsideration and entrance of the amendment in the application are respectfully requested.

Art-Based Rejections

Claims 1-3, 5, 7-12 and 14-19 were rejected under 35 U.S.C. § 102(e) over U.S. Patent No. 6,907519 B2 (Desoli); Claims 4, 13, 20 and 21 were rejected under 35 U.S.C. § 103(a) over Desoli in view of U.S. Patent No. 5,613,063 (Eustace). Applicant respectfully traverses the rejections and submits that the claims herein are patentable in light of the clarifying amendments above and the arguments below.

The Desoli Reference

Desoli is directed to emulation of an original computer system on a host computer system. Emulated code 116 can be integrated with native code 118 and processed in emulator 102. Emulation occurs only through emulator module 110. Native code interceptor module 108 is configured to detect native code 118 inserted within emulated code 116 and to execute the native code 118 on hardware 106 without the need for emulation. Therefore, emulation system 100 executes native code on hardware 106 without emulation. Emulation module 110 emulates the hardware of an emulated system such that from the perspective of a program executed by emulation system 100, emulated module 110 performs all of the actions that the original hardware would have performed natively if it were not in fact being emulated on a host computer system. In a media-based application, emulated

code 116 is executed by emulation system 100 such that emulated code 116 is hardware emulated while native code 118 is executed natively.

Native code 118 may be executed via a dynamic execution layer interface (DELI) 104, which is a generic software layer that runs between the emulator 102 and hardware 106 to provide services such as code optimization and code caching through an Application Program Interface (API). DELI also can operate in a transparent mode that controls an executing program such that the executing program is unaware that it is not executing directly on the computer hardware *(See Desoli: FIG.1-3 and 5; Col. 1, lines 15-18; Col. 3, line 57 – Col. 4, line ; Col. 4, lines 14-19, 48-51 and 61-65; Col. 5, lines 23-40; Col. 6, lines 47-63; Col. 16, lines 1-3).*

The Eustace Reference

Eustace is directed to monitoring memory accesses using special values stored as contents of the memory location themselves and a table of write tags *(See Eustace; Col. 3, lines 21-25).*

The Claims are Patentable Over the Cited References

The present application is generally directed to a method of executing a program written in a programming language which is executed with an interpreter having a native code calling function.

As defined by independent Claim 1, a computer readable medium encoded with an interpreter executes a program written in a programming language in cooperation with a processor. The interpreter includes a module that calls a native code and a native code emulator that executes the native code through hardware emulation. A determination module determines whether a target code in a program to be executed is an interpreter code or the native code. When the target code is

determined to be the native code, the native code emulator processes the native code through hardware emulation. When the target code is determined to be the interpreter code, the native code emulator does not process the interpreter code.

The applied references do not disclose or suggest the features of the present invention as defined by independent Claim 1. In particular, the applied references do not disclose or suggest, "a native code emulator that executes the native code through hardware emulation," as required by independent Claim 1.

As pointed out in the "Response to Arguments" in the last Office Action, Desoli teaches in column 4, lines 14-19 that, "emulation module 110 emulates the hardware of an emulated system. Accordingly, emulation module 110, from the perspective of a program executed by the emulation system 100, performs all of the actions that the original hardware would have performed during native execution of the program."

In this example, Desoli teaches the general principle for emulating code from an original computer system such that the original code appears to be executed on original hardware natively, when in fact it is executed on a host computer system. This passage does not suggest that Desoli performs execution of native code, as the Examiner alleges. Instead, emulation module 110 acts to execute emulated code as if the program was executed natively on the original hardware system. Emulation system 100 executes code written for an original computer system that is incompatible with the host computer system. In other words, Desoli merely teaches emulating original hardware system code (emulated code) and does not disclose or suggest emulating native code.

Furthermore, Applicant assumes this passage was cited for the disclosure of "native execution." However, from the context of the entire passage, the phrase "native execution" clearly refers to the emulated code that is native to the original system hardware and cannot be construed to read on applicant's native code.

Moreover, Desoli discloses that emulated code 116 can be integrated with native code 118 and processed in emulator 102. However, the actual process of emulation occurs only through emulator module 110. Although emulation system 100 executes both native and emulated code, only emulated code is executed by hardware emulation using emulation module 110. Native code is executed on hardware 106 without emulation *(See Desoli; FIG. 1 and Col. 4, lines 61-64).*

To further clarify this feature, Desoli teaches that in a media-based application, native hardware 106 is more efficient than an emulated system at executing native code. Accordingly, emulated code 116 is executed by hardware emulation while native code 118 is executed natively. Consequently, the overall execution speed is improved *(See Desoli; Col. 5, lines 23-40).* Thus, emulation system 100 does not emulate both native and emulated code. Each type of code is identified before execution and is executed through the appropriate means, namely hardware 106 or emulation module 110.

Moreover, Desoli discloses that native code 118 may be executed via a dynamic execution layer interface (DELI) 104 *(See Desoli; Col. 4, lines 64-65 and Col. 3, line 57 – Col . 4, line 7).* DELI is a generic software layer that runs between the emulator 102 and hardware 106 that provides services unrelated to emulation, such as code optimization and code caching. DELI provides an application program interface (API) that allows code optimization and code caching *(See Desoli; FIG. 3 and 5, Col. 6, lines 47-63 and Col. 14, lines 1-3).* DELI further teaches a transparent mode that, "takes control of an executing program in a manner in which the executing program is unaware that it is not executing directly on computer hardware," *(See Desoli; Col. 4, lines 5-8).* Importantly, the transparent mode does not provide emulation, but merely adds a control layer that is "transparent" to the executed program. Thus, DELI does not execute or emulate

code, but merely controls the program executed by either emulator 102 or hardware 106.

Furthermore, Applicant submits that since emulator 102 provides emulation, there is no reason for DELI to act as an emulator. Although DELI interfaces with the emulation system, DELI does not execute or emulate code, but merely provides additional services useful for emulated code execution or native code execution.

Even broadly interpreted, code caching is not an emulation process. Applicant submits that it makes no sense for DELI to act as an emulator since Desoli teaches the virtues of natively executing native code without emulation to increase efficiency. Doing so goes against the stated advantages of Desoli's invention such that one of ordinary skill would not think to implement DELI as an emulator of native code. However, by providing code caching and optimization, DELI would increase the efficiency of native code execution on hardware 106 *(See Desoli; Col. 5, lines 23-40).*

In summary, executed code is merely processed through an additional software layer that optimizes code and provides code caching. The code caching and optimization of Desoli is clearly not an emulation process. Thus, DELI provides ancillary services unrelated to hardware emulation such that native code is not emulated by DELI.

In contrast, the present invention requires a native code emulator that executes native code through hardware emulation. Emulating native code allows the present invention to detect illegal memory accesses and allows an execution state of the code to be saved, features not shown in the conventional solutions *(See Specification; Page 5, lines 2-4 and Page 3, lines 19 – Page 4, line 17).* Desoli merely executes native code without hardware emulation either directly by hardware 106 or via DELI 104.

Thus, Desoli does not disclose or suggest this feature of the present invention as required by independent Claim 1. The ancillary references do not remedy the deficiencies of Desoli.

Since the applied references fail to disclose, teach or suggest the above features recited in independent Claim 1, those references cannot be said to anticipate nor render obvious the invention which is the subject matter of that claim.

Accordingly, independent Claim 1 is believed to be in condition for allowance and such allowance is respectfully requested.

Applicant respectfully submits that independent Claims 11 and 16 are allowable for at least the same reasons as discussed above with reference to independent Claim 1 and such allowance is respectfully requested.

The remaining claims depend either directly or indirectly from independent Claims 1, 11 and 16 and recite additional features of the invention which are neither disclosed nor fairly suggested by the applied references and are therefore also believed to be in condition for allowance.

Conclusion

Applicant believes the foregoing amendments comply with requirements of form and thus may be admitted under 37 C.F.R. § 1.116(b). Alternatively, if these amendments are deemed to touch the merits, admission is requested under 37 C.F.R. § 1.116(c). In this connection, these amendments were not earlier presented because they are in response to the matters pointed out for the first time in the Final Office Action.

Lastly, admission is requested under 37 C.F.R. § 1.116(b) as presenting rejected claims in better form for consideration on appeal.

In view of the foregoing, it is respectfully submitted that the application is in condition for allowance. Reexamination and reconsideration of the application, as amended, are requested.

If for any reason the Examiner finds the application other than in condition for allowance, the Examiner is requested to call the undersigned attorney at the Los Angeles, California telephone number (310) 785-4721 to discuss the steps necessary for placing the application in condition for allowance.

If there are any fees due in connection with the filing of this response, please charge the fees to our Deposit Account No. 50-1314.

Respectfully submitted,
HOGAN & HARTSON L.L.P.

Date: October 1, 2007

By: 

Dariush G. Adli
Registration No. 51,386
Attorney for Applicant

1999 Avenue of the Stars, Suite 1400
Los Angeles, California 90067
Phone: 310-785-4600
Fax: 310-785-4601